Building a Simple AI Model with Python and TensorFlow

Overview

This document introduces building a basic machine learning model using Python and TensorFlow. We'll cover a simple neural network for classifying images from the MNIST dataset, a standard dataset of handwritten digits. The model will be trained to recognize digits, which is a common introductory project in AI development.

1. Setup and Installation

First, we need to install the required libraries:

```
pip install tensorflow numpy matplotlib
```

Once the necessary libraries are installed, we'll import them and load the MNIST dataset.

```
import tensorflow as tf

from tensorflow.keras import layers, models

import numpy as np

import matplotlib.pyplot as plt


# Load the MNIST dataset

(x_train, y_train), (x_test, y_test) =
tf.keras.datasets.mnist.load_data()


# Normalize the data

x_train, x_test = x_train / 255.0, x_test / 255.0
```

2. Model Architecture

We'll create a simple neural network with two dense layers. The first layer will have 128 units, and the second layer will output the probability of the 10 digits (0-9).

```
# Build the model

model = models.Sequential([

    layers.Flatten(input_shape=(28, 28)),  # Flatten the 28x28
images into a 1D array

    layers.Dense(128, activation='relu'),  # Fully connected layer
with ReLU activation

    layers.Dropout(0.2),  # Dropout to reduce overfitting
```

```
    layers.Dense(10, activation='softmax')  # Output layer with 10
neurons (one for each digit)
])


# Compile the model
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

3. Training the Model

We'll now train the model on the training data. The model will use the test data to evaluate its performance after each epoch.

```
# Train the model
model.fit(x_train, y_train, epochs=5)
```

4. Evaluating the Model

After training, we evaluate the model on the test set to check how well it generalizes to new, unseen data.

```
# Evaluate the model
test_loss, test_acc = model.evaluate(x_test, y_test, verbose=2)
print(f"Test accuracy: {test_acc}")
```

5. Visualizing Predictions

To visualize the model's predictions, we'll use Matplotlib to display an image and its predicted label.

```
# Make predictions
predictions = model.predict(x_test)


# Plot the first image in the test set and show its predicted label
plt.imshow(x_test[0], cmap=plt.cm.binary)
plt.title(f"Predicted Label: {np.argmax(predictions[0])}")
plt.show()
```

6. Conclusion

This document demonstrated how to build a basic neural network for digit classification using TensorFlow. Although this is a simple model, it introduces key concepts in AI and machine learning, including data preprocessing, model architecture, training, and evaluation.